

# Bases de Datos Relacionales con Base de OpenOffice y consultas SQL para Tecnología de la Información

1 Intro.....	1
2 BD relacionales con OO Base.....	2
2.1 Crear Base de Datos Tienda.....	2
3 Consultas con SQL.....	6
3.1 Consulta 1 SELECT *.....	7
3.2 Consulta 2.....	7
3.3 Consulta 3 WHERE.....	7
3.4 Consulta 4.....	8
3.5 Consulta 5, 5b Operadores Lógicos AND/OR/NOT.....	8
3.6 Consulta 6.....	9
3.7 Consulta 7 LIKE.....	9
3.8 Consulta 8.....	10
3.9 Consulta 9a, 9b Tablas Combinadas.....	11
3.10 Consulta 10.....	12
3.11 Consulta 11 Combinar 3 Tablas.....	12
3.12 Consulta 12 Final.....	13
3.13 Consulta 13 Extra.....	13
4 Trabajo final: db_musica.....	13
5 OUTRO.....	15

## 1 Intro

Como ya sabes las **bases de datos (BD)** son la mejor forma de almacenar y trabajar con datos que tenemos. Las utilizamos continuamente cuando navegamos por Internet al buscar una página con Google, al conectarnos a nuestra cuenta de correo, al enviar un formulario para darnos de alta en cualquier servicio,...normalmente los datos se almacenan o leen de una BD.

Recuerdas que en una BD podemos tener diferentes **objetos** como **tablas**, **formularios** y **consultas** entre otros. Todos los datos en una tabla se organizan en **campos** (columnas) y en **registros** (filas). Para visualizar los registros de una tabla normalmente se diseñan formularios que permiten también introducir nuevos registros en la BD. Pero, la verdadera importancia de las bases de datos reside en la posibilidad de realizar consultas que nos seleccionan parte de los datos de una BD que cumplen alguna condición/es. Todo esto lo venimos aprendiendo en el instituto desde 3º ESO. Así que...

¿Que vamos a aprender nuevo?

1. **Bases de Datos Relacionales**, esto simplificando mucho implica que existen unas restricciones a la hora de insertar nuevos registros en una

tabla, de tal forma que sólo se permiten introducir los datos si cumplen algunos requisitos. Por ejemplo, en un zoológico no podríamos asignar a un animal determinado un cuidador que no fuese ya empleado del zoo.

2. **Consultas con SQL.** El Lenguaje de Consultas Estructurado o SQL es un lenguaje empleado por la mayoría de las BD (MySQL, Oracle, Microsoft SQL Server, etc).
3. Manejo sencillo de **MySQL** un gestor de BD con arquitectura cliente-servidor muy empleado con los servidores Apache de páginas Web. Trabajaremos con una interfaz gráfica muy completa **phpMyAdmin**, donde crearemos tablas y haremos consultas con SQL.
4. **Diseño de Páginas Web con PHP y MySQL** combinados para presentar tablas y consultas de nuestra BD.

Vamos a dividir este curso en dos partes, en la primera trataremos las BD en modo local (**Apartados 1 y 2**) mientras que en la segunda parte trabajaremos con aplicaciones cliente/servidor desde navegadores web (**Apartados 3 y 4**).

Como ves, vamos a dar un largo paseo...

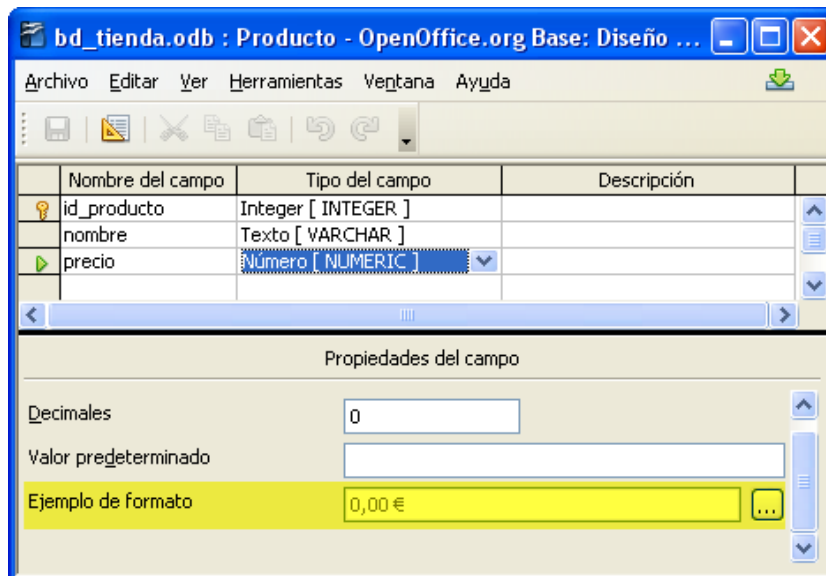
## 2 BD relacionales con OO Base


Empezaremos con una aplicación de la OpenOffice llamada Base, por la facilidad de crear tablas y de realizar consultas SQL y por su gratuidad. La puedes descargar de <http://es.openoffice.org/>

### 2.1 Crear Base de Datos Tienda

Los primeros pasos son muy sencillos, vamos a crear una BD que gestione las ventas de una tienda de informática. Nuestra BD tendrá tres tablas, una de los productos que vendemos, otra de los clientes que nos compran productos y otro de las compras que realicen los clientes.

1. Abre la aplicación gestora de bases de datos de OpenOffice llamada **Base** y crea una nueva BD llamada "**bd\_tienda**". En las opciones dile que NO queremos registrarla en openoffice.org, y que después de guardarla queremos abrirla para editar.
2. Selecciona el objeto Tablas y luego la Tarea **Crear Tabla en vista diseño** para definir los campos de la primera tabla que llamaremos **Producto** (¡ muy atento a las mayúsculas/minúsculas !). Inserta los nombres de los campos y tipos de datos que contienen, tal y como aparecen en la captura.



**Detalle:** ¡Fíjate que el formato del precio es tipo moneda € ! Y por supuesto en que la **clave primaria**  está en el campo **id\_producto**, que además es un campo **AUTONUMÉRICO**

- Introduce luego los registros en la tabla **Producto** como aparecen en la captura. Estos serán los artículos que podrán comprar los usuarios de la tienda.

id_producto	nombre	precio
1	PC	630,00 €
2	ratón	7,00 €
3	impresora	79,00 €
4	scanner	45,00 €
5	plotter	235,00 €
6	portátil	690,00 €
7	cámara web	24,00 €
8	auriculares	4,00 €
9	altavoces	20,00 €
10	micrófono	5,00 €

- Ahora, siguiendo el mismo método crea las estructuras de la tabla **Usuario**, con los campos de datos y las claves primarias que te indico a continuación entre paréntesis:

id_usuario	nombre	cuenta
1	Antonio	1234
2	Marta	4536
3	Petronio	6748
4	Javier	1123
5	Monica	2233

**Usuario** ( **id\_usuario** (Integer, Clave Primaria), **nombre** (Texto-VARCHAR), **cuenta** (Integer)).

Como puedes entender, la tabla **Usuario** contiene a los clientes que compran en nuestra tienda e incluye entre otros datos su número de cuenta.

Introduce los registros de nuestros usuarios como se muestra en la captura.

5. La tercera tabla **Carrito** contiene los pedidos de uno en uno que hace un determinado usuario. Crea su estructura con los siguientes campos:

**Carrito** (**id\_compra** (Integer, Clave Primaria, Autonumérico), **id\_usuario**(Integer), **id\_producto**(Integer))

6. **Relaciones** entre las tres tablas.

Ya tenemos creadas todas las tablas, pero aún no estamos preparados para introducir pedidos en el carrito. Está claro que los registros en la tabla **Carrito** deben cumplir algunas condiciones antes de ser guardados: sería ilógico que se hiciese un pedido de un producto inexistente o que, aunque sí tuviésemos en la tienda un determinado producto, no existiese el usuario que lo ha pedido. Estas dos son las relaciones entre tablas que vamos a establecer.

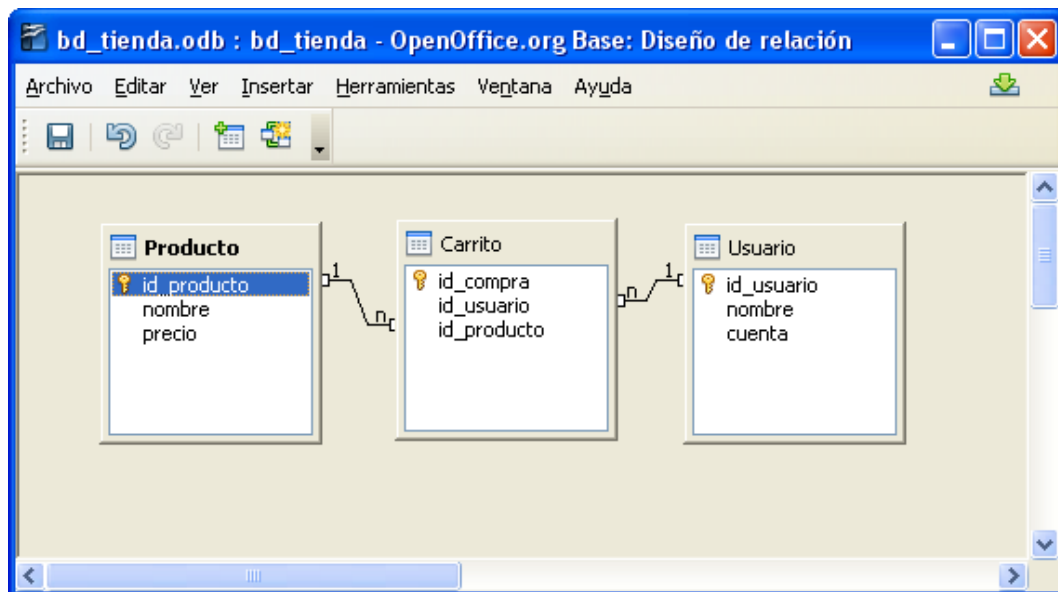
**Profundización:** Otra forma de entender las relaciones entre dos tablas es que en la “**tabla hijo**” no se insertarán registros que no coincidan con los de la “**tabla padre**” en el campo que las vincula.

*“No se permiten hijos sin padre, huérfanos, pero sí se permiten padres sin hijos.”*

**Detalle:** Sólo te diré una cosa importante sobre las relaciones, **sólo se permiten las relaciones entre campos con el mismo tipo de datos** de dos tablas distintas.

Las relaciones se crean desde el **menú Herramientas/Relaciones**, donde agregamos las tres Tablas una a una.

7. Primera relación entre las tablas **Producto** y **Carrito**, para crearla pinchamos en el campo **Producto.id\_producto** y arrastramos hasta **Carrito.id\_producto**. Veremos cómo se crea una línea continua entre ambos campos. La tabla **Producto** es la llamada “**tabla padre**” y la tabla **Carrito** es la “**tabla hijo**”.
8. Segunda relación entre las tablas **Usuario** y **Carrito**, para crearla pinchamos en **Usuario.id\_usuario** y arrastramos hasta **Carrito.id\_usuario**. Aquí la “**tabla padre**” es **Usuario** y la “**tabla hijo**” es “Carrito”.



Si lo hemos hecho bien, debemos ver las dos relaciones creadas entre las tres tablas como aparece en la siguiente captura. Para modificar las propiedades de una relación hacemos doble clic en su línea y marcamos las casillas como se ve en la captura, esto lo hacemos en ambas relaciones.

The 'Relaciones' dialog box shows the relationship between 'Carrito' and 'Producto'. The 'Campos implicados' section shows 'id\_producto' from both tables. The 'Opciones de actualización' section has 'Actualizar cascada' selected. The 'Opciones de eliminación' section has 'Eliminar cascada' selected.

Carrito	Producto
id_producto	id_producto

Opciones de actualización:

- ☐ Ninguna acción
- ☒ Actualizar cascada
- ☐ Poner null
- ☐ Predeterminar

Opciones de eliminación:

- ☐ Ninguna acción
- ☒ Eliminar cascada
- ☐ Poner null
- ☐ Predeterminar

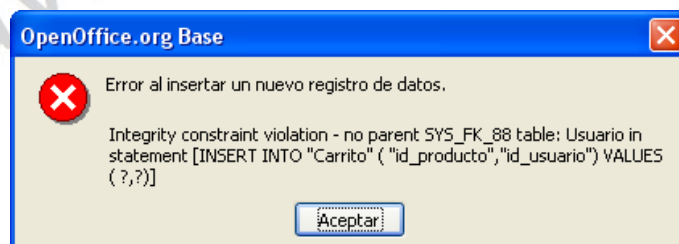
Buttons: Aceptar, Cancelar, Ayuda

9. Ya podemos insertar los pedidos en el carrito, tal y como se ve en la captura:

	id_compra	id_usuario	id_producto
0		3	1
1		3	2
2		1	1
3		2	3
6		4	3
7		3	9
8		3	8
9		5	6
10		5	7
11		2	4
12		4	4
<Campo autor			

**Detalle: ¡¡ Fíjate que no existen los códigos de id\_compra 4 y 5!!!**

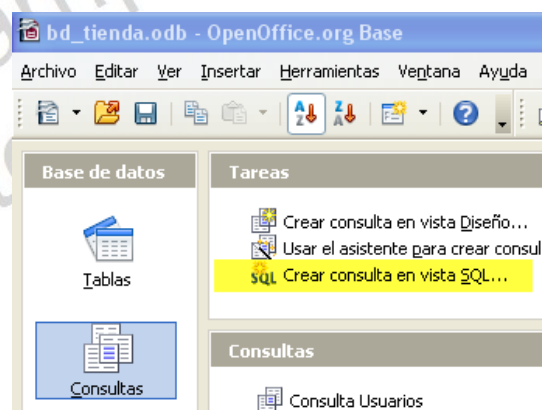
Para comprobar que efectivamente funcionan las relaciones que hemos definido entre las tablas basta con intentar insertar en la tabla **Carrito** un registro de un **id\_usuario** que no exista en la tabla **Usuario** por ejemplo el nº 34, el sistema nos informará y prohibirá guardar el registro, tal y como se ve en la captura:



Con lo explicado hasta ahora hemos creado una sencilla BD relacional y estamos preparados para realizar las consultas en SQL.

### 3 Consultas con SQL

Seguro que recuerdas que una vez creadas las tablas, podemos seleccionar datos mediante consultas. En las aplicaciones gestoras de BD como Base u OpenOffice hay una forma sencilla de crear consultas en **Modo Vista Diseño**, no es objetivo de este curso explicarlas, son muy simples e intuitivas. Debajo de las consultas en vista diseño aparece **Crear consulta en vista SQL** como se muestra en la captura:



Seleccionaremos esta forma de crear consultas y se abrirá una ventana donde escribiremos los comandos de la consulta.

El comando más importante es **SELECT**, y su sintaxis básica es:

```
SELECT "campo1", "campo2", "campo3"  
FROM "tabla"  
WHERE "condición"
```

Hemos dispuesto el código en tres líneas por claridad, pero podría estar en sólo una todo seguido.

La sentencia **SELECT** permite seleccionar los campos cuyos datos queremos filtrar, **FROM** indica el nombre de la tabla o tablas donde pertenecen los campos y **WHERE** alguna condición que cumplirán los datos que mostraremos de los campos seleccionados.

**Detalle:** ¡¡¡ En SQL hay que respetar las MAYÚSCULAS/minúsculas o te mostrará continuos avisos de "Sintaxis Errónea" !!!

Todo es más sencillo de lo que parece, sólo tenemos que empezar a crear consultas...

### 3.1 Consulta 1 **SELECT \***

Selecciona **todos los campos** y **todos los registros** de la tabla **Usuario** mediante este código:

```
SELECT * FROM "Usuario"
```

Guárdalo como Consulta1, el resultado se verá igual que la tabla **Usuario**.

**Detalle:** El asterisco **\*** es el comodín y significa cualquier cosa.

### 3.2 Consulta 2

Ahora tú haz una consulta de todos los campos de la tabla **Producto** y guárdala como **Consulta2**

### 3.3 Consulta 3 **WHERE**

Vamos a empezar a usar la sentencia condicional **WHERE**. Por ejemplo, seleccionaremos los registros de los campos **nombre** y **precio** de la tabla **Producto** que tengan un precio menor de 25 euros. El código sería:

```
SELECT "nombre", "precio" FROM "Producto" WHERE  
"precio" < 25
```



La captura muestra la tabla consulta3:

nombre	precio
ratón	7
cámara web	24
auriculares	4
altavoces	20
micrófono	5

De la misma forma podemos usar otros operadores:

<	Menor que
<=	Menor o igual que
>	Mayor que
>=	Mayor o igual que
=	igual

**Detalle:** Si en vez de un valor numérico (25) quisiéramos usar una palabra, habría que entrecomillarla con **comillas simples** ('ratón').

### 3.4 Consulta 4

Ahora tú haz una consulta de todos los pedidos del **id\_usuario** 3. Deben aparecer los campos **id\_compra**, **id\_usuario** e **id\_producto** de la tabla **Carrito**.

La captura muestra la tabla consulta4.

id_compra	id_usuario	id_producto
1	3	1
1	3	2
7	3	9
8	3	8

### 3.5 Consulta 5, 5b Operadores Lógicos AND/OR/NOT

Vamos a empezar a usar AND, los demás operadores AND y NOT se emplearían igual. Por ejemplo, si queremos saber si el **usuario 3** ha comprado el **producto 2** introduciremos el siguiente código. El código sería:

```
SELECT "id_usuario", "id_producto" FROM "Carrito"
WHERE "id_usuario" = 3 AND "id_producto"=2
```



La captura muestra la tabla consulta5, efectivamente el usuario 3 **SÍ** ha comprado el producto 2.

id_usuario	id_producto
3	2

**Consulta 5b** . Uso de **NOT**, consulta de los **usuarios** que **NO** hayan pedido el **producto 2**, sería:

```
SELECT "id_usuario", "id_producto" FROM "Carrito"
WHERE NOT "id_producto"=2
```

Como se ve en la captura de la Consulta5b.

id_usuario	id_producto
3	1
1	1
2	3
4	3
3	9
3	8
5	6
5	7
2	4
4	4

### 3.6 Consulta 6

Ahora tú haz una consulta de todos los usuarios que hallan pedido los productos 3 o 4. Deben aparecer los campos **id\_usuario** e **id\_producto** de la tabla **Carrito** y usar **OR** en lugar de AND.

La captura muestra la tabla consulta6.

id_usuario	id_producto
2	3
4	3
2	4
4	4

### 3.7 Consulta 7 LIKE

Podemos buscar datos muy detallados con el comando LIKE. Si buscamos en la tabla **Producto** aquellos cuyo nombre empieza por "a", en un explorador

de Windows teclearíamos “a\*” ya que “\*” significa “cualquier cosa”. Pero con SQL empleamos “%” en su lugar. De igual forma en Windows “?” significa un sólo carácter comodín, y en SQL se usaría el guión bajo “\_”.

*Detalle: En la siguiente tabla se muestran algunos ejemplos:*

Búsqueda	Explorador de Windows	SQL
Palabras que acaben en “os”	*os	'%os'
Palabras que empieza por “ojer” y tengan después 2 caracteres cualesquiera (ojer-as vale, pero ojer-oso no)	ojer??	'ojer__'
Palabras que contengan “eso”	*eso*	'%eso%'

Vamos a seleccionar los registros de los campos **nombre** y **precio** de la tabla **Producto** cuyo nombre contenga el carácter “e” en penúltima posición. El código sería:

```
SELECT "nombre", "precio" FROM "Producto"
WHERE "nombre" LIKE '%e_'
```

nombre	precio
scanner	45
plotter	235
cámara web	24
auriculares	4
altavoces	20

La captura muestra la tabla consulta 7.

### 3.8 Consulta 8

Ahora tú haz una consulta de todos registros de los campos **nombre** y **cuenta** de la tabla **Usuario** cuyo **nombre** contenga una “n” en antepenúltima posición, esto es la 3ª contando desde el final.

La captura muestra la tabla consulta8.

nombre	cuenta
Antonio	1234
Petronio	6748

### 3.9 Consulta 9a, 9b Tablas Combinadas

Si queremos mostrar las dos tablas **Usuario** y **Carrito** completas, probaríamos incluir ambas tablas en la consulta SELECT.

```
SELECT * FROM "Carrito", "Usuario" WHERE "Usuario"."id_usuario" = "Carrito"."id_usuario"
```

Consulta9a - bd\_tienda - OpenOffice.org Table Data View

Archivo Editar Ver Insertar Herramientas Ventana Ayuda

	id_compra	id_usuario	id_producto	id_usuario1	nombre	cuenta
0	3	1	3	Petronio	6748	
1	3	2	3	Petronio	6748	
2	1	1	1	Antonio	1234	
3	2	3	2	Marta	4536	
6	4	3	4	Javier	1123	
7	3	9	3	Petronio	6748	
8	3	8	3	Petronio	6748	
9	5	6	5	Monica	2233	
10	5	7	5	Monica	2233	
11	2	4	2	Marta	4536	
12	4	4	4	Javier	1123	

Registro 1 de 11

Pero, como ves iii aparece duplicado el campo **id\_usuario** que comparten ambos !!!!

Hay una solución para eliminar duplicidades, si seleccionamos todos los campos de la tabla **Carrito** con la sentencia "Carrito.\*" pero en la otra tabla, seleccionamos uno a uno todos sus campos salvo **Carrito.id\_usuario** que se repite en ambas tablas:

```
SELECT "Carrito".*, "Usuario"."nombre", "Usuario"."cuenta" FROM "Carrito", "Usuario" WHERE "Usuario"."id_usuario" = "Carrito"."id_usuario"
```

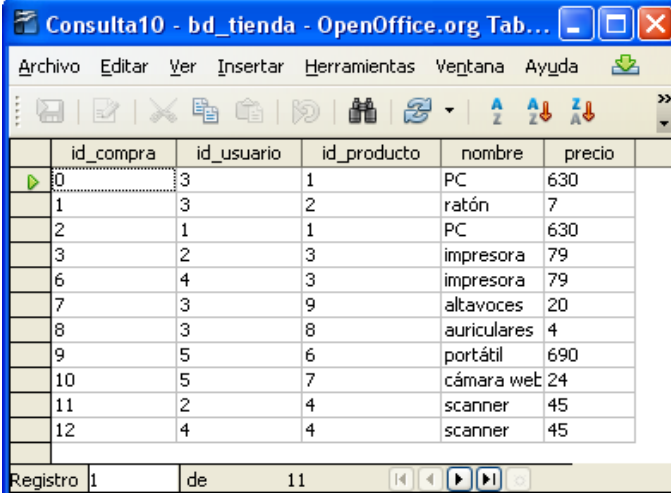
Crea esta nueva consulta y guárdala como **Consulta 9b**.

Consulta9b - bd_tienda - OpenOffice.org Table D...					
Archivo Editar Ver Insertar Herramientas Ventana Ayuda					
id_compra id_usuario id_producto nombre cuenta					
0	3	1	Petronio	6748	
1	3	2	Petronio	6748	
2	1	1	Antonio	1234	
3	2	3	Marta	4536	
6	4	3	Javier	1123	
7	3	9	Petronio	6748	
8	3	8	Petronio	6748	
9	5	6	Monica	2233	
10	5	7	Monica	2233	
11	2	4	Marta	4536	
12	4	4	Javier	1123	
Registro 1 de 11					

### 3.10 Consulta 10

Ahora tú haz una consulta que combine las tablas **Carrito** y **Producto** sin que aparezcan campos duplicados.

La captura muestra la tabla consulta10.

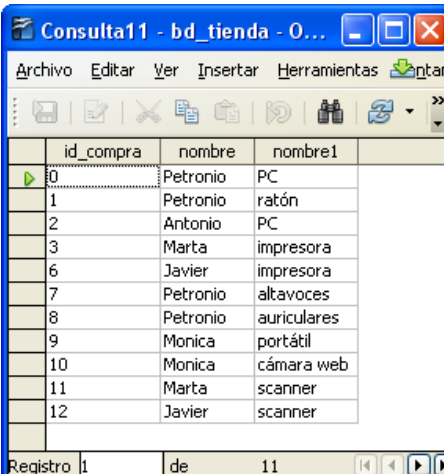


	id_compra	id_usuario	id_producto	nombre	precio
0	3	1	1	PC	630
1	3	2	2	ratón	7
2	1	1	1	PC	630
3	2	3	3	impresora	79
6	4	3	3	impresora	79
7	3	9	9	altavoces	20
8	3	8	8	auriculares	4
9	5	6	6	portátil	690
10	5	7	7	cámara web	24
11	2	4	4	scanner	45
12	4	4	4	scanner	45

### 3.11 Consulta 11 Combinar 3 Tablas

Hasta ahora, en ninguna consulta hemos mostrado juntos los campos **id\_compra**, **nombre** del **producto** comprado y **nombre** del **usuario** que lo compró. Para ello hay que pensar un poco, pero realmente TU ya no necesitas que YO te muestre el código...

Primero selecciona los campos que queremos mostrar y sus tablas e imponemos dos condiciones: que sólo se filtren los **id\_usuarios** que hayan comprado algo y los **id\_productos** que se hayan vendido, esto es que existan en la tabla **Carrito**.



	id_compra	nombre	nombre1
0		Petronio	PC
1		Petronio	ratón
2		Antonio	PC
3		Marta	impresora
6		Javier	impresora
7		Petronio	altavoces
8		Petronio	auriculares
9		Monica	portátil
10		Monica	cámara web
11		Marta	scanner
12		Javier	scanner

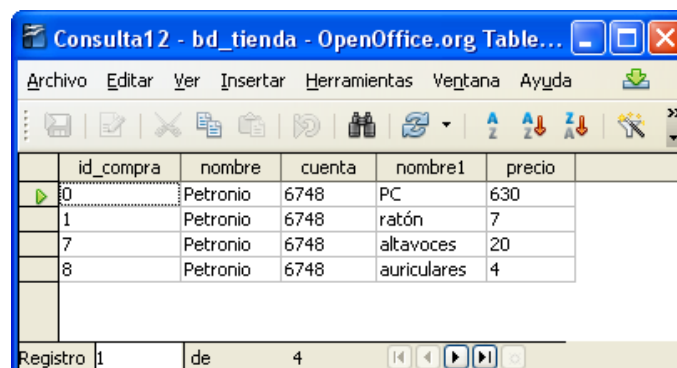
```
SELECT "id_compra", "Usuario"."nombre", "Producto"."nombre"
FROM "Carrito", "Usuario", "Producto"
WHERE "Carrito"."id_usuario" = "Usuario"."id_usuario"
```

AND "Carrito"."id\_producto" = "Producto"."id\_producto"

### 3.12 Consulta 12 Final

Ahora, como ejercicio final te pido que me muestres un consulta donde aparezcan juntos los campos **id\_compra** del **Carrito**, **nombre** y **precio** del **Producto** comprado y **nombre** y **cuenta** del **Usuario** que lo compró, pero sólo del usuario llamado **Petronio**.

**Detalle:** cuando incluyas en las condiciones a Petronio, no olvides meterlo entre comillas simples : 'Petronio'

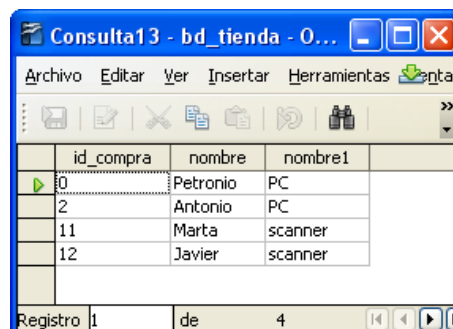


	id_compra	nombre	cuenta	nombre1	precio
0	Petronio	6748	PC	630	
1	Petronio	6748	ratón	7	
7	Petronio	6748	altavoces	20	
8	Petronio	6748	auriculares	4	

Registro 1 de 4

### 3.13 Consulta 13 Extra

Como final para los que verdaderamente NO han desaprovechado COMPLETAMENTE el curso... ¿podrías mostrarme los **nombres** y códigos de **compra** de los clientes que han comprado un **scanner** o un **PC**?



	id_compra	nombre	nombre1	
0		Petronio	PC	
2		Antonio	PC	
11		Marta	scanner	
12		Javier	scanner	

Registro 1 de 4

## 4 Trabajo final: db\_musica

Te pido ahora que crees una BDR sobre la música preferida de tus amigos. Tendrá **tres tablas: artistas, discos y amigos**.

**Buscarás los datos que necesites en Internet y serán datos verdaderos. Deberás decidir por ti mismo los tipos de datos para cada campo e indicar sus propiedades, si es un campo requerido o no, su formato, etc., según tu coherencia**

y sentido común.

1. Tabla **artistas (con 10 registros)**, con los siguientes campos: **nombre** (llave primaria), **año de fundación**, **nacionalidad** y **nº de discos** oficiales publicados hasta hoy.
2. Tabla **discos (con 20 registros)**, con lo campos: **título** (llave primaria), **artista**, **año** de publicación, **nº de canciones** y si ha sido o no **disco de oro**.
3. Tabla **amigos (con 10 registros)**, con los campos: **nombre** (llave primaria) y **disco preferido**.

Las **relaciones** entre tablas deben tener en cuenta lo siguiente:

- Sólo se admitirán en la tabla “discos” aquellos nombres de artistas que ya existan en la tabla “artistas”. Tabla padre = “artistas”, Tabla hijo=“discos”.
- Solo se admitirán como títulos de disco preferidos de mis amigos aquellos títulos que existan en la tabla “discos”. Tabla padre=“discos”, Tabla hijo=“amigos”.

Una vez diseñadas las relaciones, inserta datos ilógicos para **comprobar** que las **restricciones funcionan**. Por ejemplo: inserta en la tabla “amigos” algún título de disco preferido que no exista y comprueba que NO te permiten insertar el registro en la tabla. Haz lo mismo en la tabla discos inventándote nombres de artistas inexistentes.

Si la aplicación OpenOfficeBase Sí te permitiese introducir estos registros erróneos.....

iiii TIENES MAL DISEÑADAS LAS RELACIONES EN LA BASE DE DATOS.!!!!

Si todo ha funcionado bien puedes continuar con las consultas...

**Consultas:** te propongo consultas sencillas (nivel +), de nivel medio ( nivel ++) y avanzadas (nivel +++).

**ii Para aprobar un examen de BDR debes hacer con soltura y sin ningún tipo de ayuda externa las consultas de niveles + y ++ !!**

Consulta nº	Nivel	Descripción
1	+	Mostrar todos los registros y campos de la tabla “artistas”
2	+	Mostrar los títulos y artistas de los todos los discos
3	+	Mostrar los discos preferidos y el nombre de mis amigos cuyos nombres contengan la letra “a”
4	+	Mostrar la fecha de fundación, nombre del artista y nº de discos

		publicados de aquellos artistas que empezaron antes del año 2000
5	++	Mostrar los títulos y artistas de los discos que hayan sido de “oro”
6	++	Mostrar el nombre de mis amigos, sus discos preferidos y si son de “oro” de aquellos discos preferidos de mis amigos que hayan sido disco de “oro”
7	++	Mostrar el título y artista de aquellos discos cuyos artistas sean de USA
8	++	Mostrar todos los registros de las tablas artistas y discos juntos pero sin duplicar los campos
9	+++	Mostrar los títulos, nº de canciones, artistas y nacionalidad de aquellos discos con menos de 12 canciones y cuyos artistas NO sean españoles
10	+++	Mostrar el nombre de mis amigos, sus títulos_preferidos, los artistas y su nacionalidad de aquellos amigos cuyo nombre contenga en segunda posición la letra “n” o la letra “o”
11	+++	Mostrar el nombre de mis amigos, sus discos preferidos, su número de canciones y el año en que se publicaron que, o hayan sido publicados después de 2005 o tengan más de 10 canciones.
12	+++	Mostrar el nombre de mis amigos, el título de sus discos preferidos y el número de discos publicados de aquellos discos preferidos cuyos artistas hayan publicado más de 7 discos oficiales
13	+++	Mostrar los campos implicados de aquellos artistas cuyos discos preferidos por mis amigos hayan sido publicados con posterioridad al 2000 y tengan más de 12 canciones
15	+++	Mostrar el título del disco y nombre del artista de aquellos artistas que sean extranjeros y que sí hayan conseguido disco de oro.

## 5 OUTRO

Evidentemente, SQL no termina aquí, hay docenas de comandos que no hemos tocado, como crear tablas, insertar, modificar, borrar, ordenar datos, campos y tablas, establecer relaciones, crear índices, copias de respaldo de la BD,... en fin, la lista sería interminable. Pero, para entrar con buen pié en la parte II del curso vale con lo aprendido para hacer Consultas SQL. Recuerda que a continuación, haremos consultas SQL a través de una página Web sobre una BD gestionada por MySQL.